

# STORM2: Process-Guided Online Dispute Resolution

Borislava I. Simidchieva, Leon J. Osterweil

August 18, 2009

## 1 Introduction

In a world where the number and variety of interactions continues to grow, the need to facilitate the resolution of disputes grows correspondingly. We note, for example, that people have used the internet to engage in increasingly broad types of interactions with each other and with vendors at a rate that continues to accelerate. And we note that most of these interactions have the potential to lead to conflict. It should be no surprise, therefore, that popular e-commerce sites such as eBay [5] and Paypal [11] incorporate automated dispute resolution facilities that clients can use to deal with disputes. Indeed eBay clients have been observed to use this facility millions of times every month.

But it is not just internet commerce sites that have been turning increasingly to the use of automation to help them deal with growing numbers of conflicts that require resolution. We note that only a very small fraction of legal disputes in the US are settled by actually going to trial. The overwhelming majority of these legal disputes are settled with the aid of Alternative Dispute Resolution (ADR) approaches, such as negotiation and bargaining.

Government dispute resolution agencies have also investigated the use of these approaches. The US National Mediation Board (NMB) is charged with the responsibility of facilitating the resolution of labor-management disputes that arise in US transportation industries (e.g. airline and railroad disputes). NMB's traditional approach to dispute resolution has been to invite disputants to a (presumably neutral) site for face-to-face negotiations around a "bargaining table." In recent years, the NMB has increasingly used ADR approaches to mediate these disputes. NMB has gone further, however, exploring the use of Online Dispute Resolution (ODR) [31], namely the use of computer and communications facilities as the basis for resolving disputes. The use of ODR can replace the physical bargaining table with a virtual forum in which the dimensions of a dispute and its possible resolutions can be explored without the need for the parties to a dispute to be physically present in the same place or at the same time, and at a much lower cost. Indeed ODR systems can even support negotiations between parties whose identities are unknown to each other, perhaps only part of the time, but perhaps at all times.

Such early successes with ODR have been encouraging and seem to be leading to more widespread adoption of this approach, but these successes have also raised questions about what the most effective role of computer support for dispute resolution processes might be. Clearly email and blogging can enable disputants to communicate with each other even though they may be separated in time and space. But the eBay example suggests that automated support can be more elaborate, more proactive, and more effective than simply allowing disputants to send email to each other. Thus, it seems interesting, for example, to consider the extent to which ODR systems might also support a mediator (either human or automated) whose role is to proactively guide and manage dispute resolution negotiations. On the other hand, there may also be disadvantages to ODR. Thus, for example, it may be undesirable for disputants to negotiate with each when they are separated in time and space. Face-to-face negotiations may offer advantages under some circumstances, although perhaps causing problems at other times. Likewise, it is not clear that anonymity is desirable at all times (indeed even at any time) during a negotiation.

Our view is that research is needed in order to address these questions, and to understand which ODR approaches, systems, and tools seem most effective in facilitating dispute resolution, and under what circumstances. The past use of some existing ODR tools is starting to create a base of experiences that is leading to such understandings. But we believe that the pace of understanding might be accelerated by experimenting with an ODR support system that is sufficiently flexible. Accordingly we have built an ODR system that can rapidly support modifications in such dimensions as the degree of mediator proactivity it supports and the degree of anonymity that it affords. This paper describes our approach to implementing a system with this desired level of flexibility, and indicates some of the evaluations that the system we have built should be able to support.

Section 2 introduces the STORM2 system by first outlining the architecture of STORM2, and describing the technologies used, then detailing the interest-based negotiation process that currently guides STORM2. Section 3 gives an overview of how the system could be used. Some implementation choices regarding the architecture of the system are explained in more technical detail in Section 4, followed by a brief review of the ongoing evaluation of the system and some preliminary findings. Section 4 also demonstrates the flexibility of the STORM2 system for accommodating different variations. Section 5 identifies and discusses some important research directions for future investigation, and Section 6 summarizes previous work in ODR and collaborations systems, as well as related work on software product lines and accommodating variation in software. Section 7 summarizes our approach and contributions.

## 2 Approach

In this section we describe the STORM2 system, which we designed to provide very flexible support for guiding disputing parties towards the resolution of their disputes. Our approach to providing flexibility is based principally on defining dispute resolution as a precisely specified process definition that can be edited, very much in the way that an executable program can be edited. In analogy to executable programs, making changes to STORM2's process definition will cause the process thereby modified to execute differently from the original version of the process. Thus, experimentation with different approaches to ODR should be facilitated because one can easily generate alternative ODR systems by creating correspondingly different process definitions.

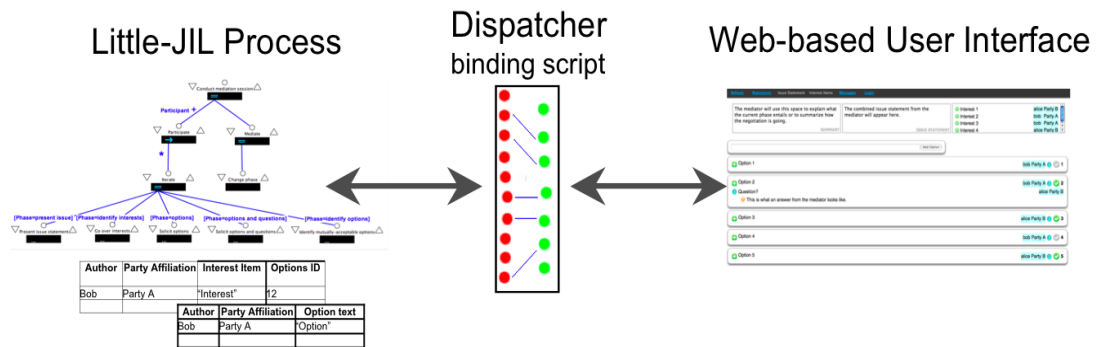


Figure 1: Three major components comprise the STORM2 system.

Figure 1 shows a very high level sketch of the architecture of STORM2. The STORM2 framework consists of a precise

and rigorous definition of a negotiation process that guides execution (shown on the left), a user interface (right), and middleware to integrate the two (center). We now provide some additional details about the STORM2 system, focusing on indicating how this approach achieves the flexibility required to support the experimentation we have suggested above. The leftmost part of Figure 1 represents the definition of a specific ODR process, defined using the Little-JIL process definition language [20]. An editor, Visual-JIL (not shown), is available to effect changes to a defined process. A Little-JIL process definition consists of three main parts, namely a coordination diagram, an artifact specification, and an agent specification. The coordination diagram is aimed mainly at specifying the coordination of the various agents, i.e. human users and automated support tools, that must collaborate in order to achieve desired outcomes. The agents, namely the mediator, participants and necessary automated support tools are defined in the agent specification of the process. Much of the coordination specification focuses on how data artifacts (e.g. position statements, opinions, and comments offered by participants) are created by some users at some times, and then routed from the users that have created these data artifacts to other users and to data management tools which require them in order to perform their own tasks. These data artifacts are specified and maintained by the artifact specification of the process. In the case of STORM2 process definitions, STORM2's users (i.e. the disputants and mediator) are asked to submit inputs at specified times, and allowed to submit their replies at other times. STORM2 is responsible for being sure that submitted inputs are routed correctly and in a timely way without violating any restrictions on confidentiality, and without violating agreements about anonymity. The right-hand box in Figure 1 represents the screens through which STORM2 users interact with the system. The implementation of this interface is based on the Tapestry framework [1], which makes the interface user-accessible online through a web browser without requiring the user to install any software. The middle box in the diagram represents a STORM2 middleware system, called the Dispatcher, which communicates requests and responses between the executing process on the left and its users through their user interfaces on the right. The Dispatcher is written in the Java programming language, and it integrates the process definition with the interface by passing messages and commands between them, based on a binding script that provides a mapping between the two. This three-component architecture, consisting of the Little-JIL process definition, the online user interface, and the Dispatcher, makes STORM2 very flexible and allows for easy modifications to accommodate the desire to create variations in the negotiation process.

## 2.1 Process Guidance

At a high level, STORM2 provides software support for performing negotiations in order to achieve dispute resolution or bargaining between two parties. To do so, it enforces the dictates of a predefined process that guides the negotiations, being able to provide important guidance to the mediator when needed. It is important to note, however, that STORM2 provides *tailored* process guidance. That is, since the way that the system executes can be changed by simply changing the process definition, STORM2 can easily be customized to support negotiation processes that can differ depending upon whatever seems appropriate. This can be done by designing and defining a custom-built negotiation process, but more efficiently by making changes and modifications to an existing process. STORM2 thus does not mandate that a mediator follow a rigid process that is unsuitable for the current situation, but only provides guidance in accordance with the specifications of a mediation process that has been designed to provide only the type of guidance that is felt to be needed, and only under circumstances that have been specified to be appropriate.

The details of any negotiation process used in a specific situation should ideally be understood and agreed to by all negotiation participants prior to their participation in a negotiation, so that the participants know how the process will coordinate their activities and agree that the plan is equitable and effective. Previous experiences of organizations such as the NMB suggest that when the participants are better aware of what to expect from the process that will guide their negotiation, they are more likely to be satisfied with the negotiation's outcomes. This means that a system such as STORM2, which is based on an explicit specification of a mediation process, may be particularly effective in encouraging participants to fully engage in a negotiation process, thereby also building trust and improving the chances of success. This process-centeredness also seems to make systems such as STORM2 particularly suitable for supporting the training of new

mediators in specific mediation processes, and in administering ODR in general. STORM2 may not provide some of the advanced features that may be found in some commercial mediation and collaboration systems [4, 9, 12, 6, 10, 13]. Instead STORM2 is based upon a conjecture that user comfort is at least as strongly based upon familiarity with, and acceptance of, the negotiation process being followed. Further, we conjecture that mediators and participants who are already familiar with the mediation approach being taken may require fewer elaborate user interface features and advanced capabilities. But such aspects of usability need to be studied and evaluated in future experiments.

## 2.2 Interest-Based Bargaining and Negotiation

Although STORM2's flexible architecture makes it a suitable basis for adaptation for use in supporting a very wide range of negotiation processes and approaches, in order to demonstrate our flexible approach to supporting ODR as clearly as possible, and also to establish a basis for initial experimentation, we have settled on one specific approach, namely Interest-Based Bargaining (IBB) [25] as the basis for the ODR processes that we initially defined and studied. We note that the NMB, in particular, uses the ideas of IBB in much of the work that it does in facilitating contract negotiations, and note further that the NMB also often uses interest-based mediation, an application of the ideas of IBB, specifically to support its work in dispute resolution and grievance mediation. Thus our initial experimentation is based upon approaches for which there has been actual practice and successful experience. In that experience there have typically been two negotiating parties (labor and management), with each party consisting of several members, or participants. In addition, each party has contained one unique member, called the party representative, whose role has been to represent, and speak for, the party.

Interest-based mediation consists of five major phases, namely identifying an issue statement, contributing interests, contributing options, contributing clarifying questions, and finally, voting for mutually acceptable options. In this paper, as in most disputes, we indeed assume that there are two negotiating parties, (e.g. labor and management). As suggested above, we believe that it is important that interest-based negotiation not be thought of as a specific fully detailed specification of a precise way to support bargaining or negotiation. Rather interest-based negotiation should be thought of as a skeleton or architecture of a family of such interest-based negotiation approaches. Thus in order to create a particular executable interest-based negotiation process it is necessary to provide considerable elaborative details to the overall five-phase interest-based negotiation framework described above. This seems to make STORM2 a particularly appropriate system to use to support interest-based negotiation, as STORM2 supports the provision and modification of such details in order to create various different interest-based negotiation processes whose specific details can differ in order to meet different needs as perceived by the disputant parties and the mediator. In order to understand this, we now provide a very high level specification of the overall interest-based negotiation approach, and then indicate how this can be elaborated in different ways to yield different negotiation processes.

Figure 2 provides a diagrammatic overview of the interest-based negotiation process. The boxes in this flowchart represent activities, and the arrows represent the flow of control between these activities. Every activity is colored to indicate who is responsible for carrying out the activity. Thus, green boxes are the responsibility of the neutral party, or the mediator, and red boxes and blue boxes are the responsibility of each of the two parties that are negotiating the dispute. As Figure 2 shows, the mediator must begin the mediation session before the negotiation can proceed. During this activity, the mediator must explain the process to the participants and get their contact information. Once the negotiation begins, the negotiating parties are asked to identify the issue that they will be negotiating. Each party is asked to separately construct a party issue statement, in the form of an open-ended question that accurately and concisely portrays the issue as the party sees it. The mediator then takes the issue statements identified by the two parties and attempts to synthesize a new, combined statement. Once the combined issue statement is enunciated, the parties are asked to identify their interests with respect to the combined issue statement. This activity is carried out by both parties together, as indicated by coloring the boxes representing such activities with the blue/red gradient in Figure 2. Each participant is encouraged to submit multiple interests, items that are important to that participant, and that the participant would like to have satisfactorily resolved at the conclusion of the negotiation session.

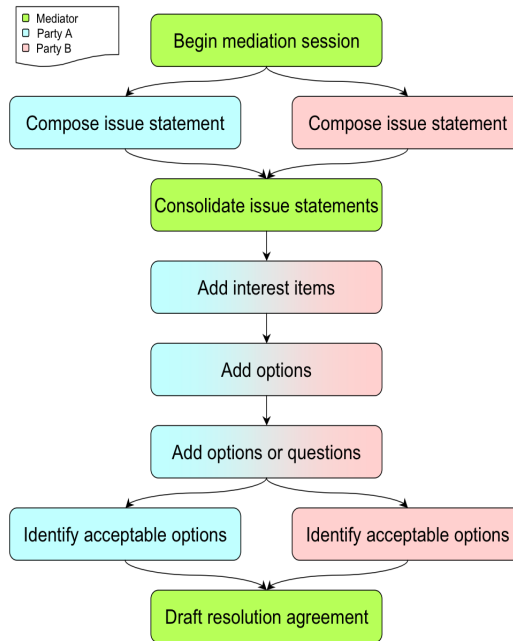


Figure 2: The interest-based negotiation approach.

Once the issue and the interests have been identified, the process calls for a focus on working toward a solution through brainstorming. Parties are asked to propose options, or ideas that may address the issue and satisfy some of the interests. After an initial period of soliciting options, the mediator allows the participants from both parties to ask clarifying questions about each other’s options, while continuing to contribute more options that may be inspired by the newly created discussion. While questions are encouraged in order to clarify ambiguities, participants are generally discouraged from commenting on the merits of a certain option in order to maintain a collaborative atmosphere and facilitate creative brainstorming. These two activities, namely the initial solicitation of options and the following solicitation of additional options along with clarifying questions, form the brainstorming component of the negotiation process that STORM2 supports. Finally, after the participants are done contributing options and asking clarifying questions, there is a brief caucus period during which each party meets separately and identifies the options that they consider acceptable. After the caucus, the party representatives from both parties vote separately for all the options that their parties have identified as acceptable. This is indicated in the flowchart by two separate activities, one for each of the parties. Once the negotiation process has completed, the mediator typically takes the options that have been deemed acceptable by both parties and uses them as the bases for drafting a resolution agreement, an activity usually carried out using a collaboration or document-sharing system not directly supported by STORM2.

Although the interest-based negotiation process may seem straightforward when described at this high level, it is important to note that many details have been omitted from Figure 2 for the sake of clarity. These omitted details are important, however, and they must be specified in order to effect the creation of a STORM2 interest-based negotiation process definition whose execution will seem comfortable and constructive to the parties. For example, it is quite possible

that either party might disagree with the combined issue statement that the mediator has composed. In this case the process must specify the details of how the parties will refine their respective issue statements, how the mediator will proceed to rephrase the combined statement to accommodate the changes, and how all of this will be iterated until both parties are satisfied. Indeed there are many different ways in which all of this might be done, but one way must be agreed upon by the parties, and then defined precisely in order to enable the interest-based negotiation process to be supported by STORM2. We note that a very broad variety of these different processes can be defined using STORM2's process definition facilities, and that the mediator and the parties are indeed expected to be participants in decisions about precisely which process is to be used. It is particularly important to note that the notation used to define these processes in STORM2 is visual, and designed to make it particularly easy for participants to understand, thereby increasing the probability that their decisions about process details will be well informed. We next describe how the STORM2 system would be used by a user in a typical negotiation scenario.

### **3 User View of STORM2**

In this section we present a high level view of STORM2 and the way it is intended to be used. Previous sections have emphasized our view that systems such as STORM2 need to be quite flexible in order to support a range of negotiation processes, but that once a particular negotiation process has been identified, it is also important that STORM2 enforce that process firmly, preventing deviations from it. Our approach to balancing the needs for both flexibility and firmness is to support customization and tailoring of process details prior to initiation of process execution, and to then support firm enforcement of those details, once they have been decided upon. Thus it is not possible to describe the functioning of STORM2 in detail, as those details may vary, by design. This section will describe the process that STORM2 is expected to enforce only at a relatively high level. We will indicate places and ways in which this "generic" STORM2 process might be tailored and customized into any of a number of more specific versions.

The processes that STORM2 is currently expected to execute comprise five major phases, as described above, usually executed in sequential order: formulating an issue statement, identifying interests, soliciting options, soliciting clarifying questions and additional options, and, finally, identify acceptable options.

In a STORM2 process the mediator is expected to determine when to progress from one phase to the next. For each different phase different users will have access to different functionality. The functionality available to the different users during each of these phases is described in more detail below. Since STORM2 negotiations may be carried out among participants who may be chronologically or geographically separated, users are expected to participate online. Because the STORM2 interface is web-based, it is easily accessible from any standard web browser with an internet connection. After a user logs into the system, he or she uses the links shown across the top of the screen to navigate the system and access the different functionality that is offered during the current phase (see Figure 3). Users use the Refresh link in the top left corner of the screen to determine if there is new content for them to view, or if new functionality has been made available by the mediator. In addition to phase-specific functionality, at any time during the negotiation users have access to the STORM2 Message Center.

#### **3.1 Message Center**

The Message Center allows users to receive broadcast or private messages from the mediator, as well as to send private messages to the mediator. Message exchange between participants is not allowed because the purpose of brainstorming is for the entire group to have access to as many ideas as possible, therefore backchannel communication is discouraged. To access the Message Center, users click on Messages (Figure 4). To read a message, a user need only click on its sender.

To send a message to the mediator, users can click on Compose, write their message in the dialog box that pops up, and then click Send (Figure 5).



Use the links above to participate in a brainstorm session.

Figure 3: The STORM2 toolbar.



Figure 4: STORM2 Message Center.

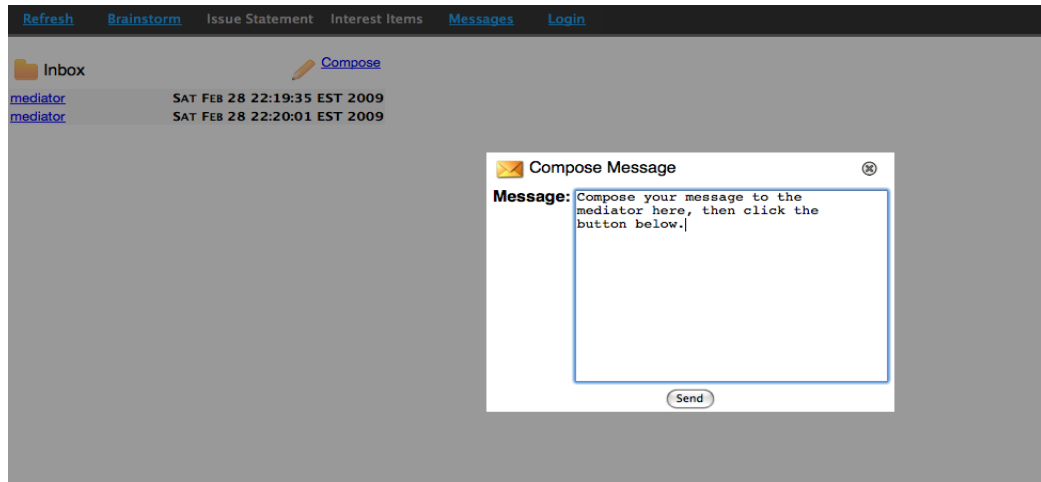


Figure 5: Sending messages.

## 3.2 Issue Statement

The first phase of the negotiation consists of formulating an issue statement. Users would only have access to this feature if the mediator has notified them that they are serving as party representatives (typically there is only one representative per party). Users who are not their party's representative are typically able to see the issue statement composed by their party representative, but are not able to compose a statement themselves. Other rules governing who can see and modify statements might be defined and then enforced using STORM2, however. During this phase, both parties must identify an issue and state it in the form of an open-ended question. A party representative can click on Issue Statement to access this feature (see Figure 6), which enables the party representative to compose a statement. Party representatives must then click on Propose New Statement when they are finished composing their submissions, at which time their statements are transmitted to the mediator. At that time the Propose New Statement button will become grayed out.

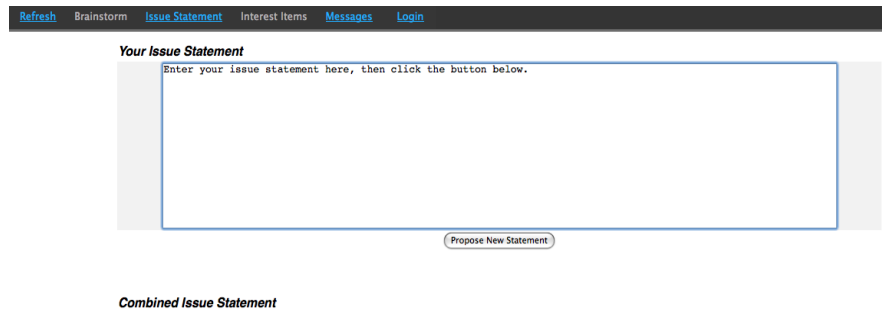


Figure 6: Proposing an issue statement.

After receiving issue statements from both party representatives, the mediator must synthesize a group issue statement to reflect both parties' suggestions. Once the mediator has composed the statement all users can view it by clicking on the Refresh link. The combined statement will then appear on the bottom of the user's screen. The precise way in which the combined statement is evolved by the joint efforts of the mediator and the parties is another example of an aspect of the negotiation process for which STORM2 can be used to specify and then enforce details. STORM2 might be used, for example, to mandate that certain parties must see and approve each new attempt to enunciate a joint statement. Or, for example, it might either mandate or forbid input from one or both of the parties during each iteration. Specific details such as these would seem to be potentially quite important to assuring that all participants in a negotiation feel sufficiently heard and empowered. Thus it seems important to have a system such as STORM2 developed and defined in such a way as to enable all concerned stakeholders to contribute towards agreement about these process details. It seems equally important to note that STORM2 is able to assure that, once agreed upon, such details will be obeyed during the execution of the negotiation.

## 3.3 Interest Items

After the mediator has formulated a group issue statement that is satisfactory to both parties, the mediator is expected to advance the negotiation to the next phase, identifying interests, which is a particularly critical phase in interest-based negotiation. In advancing the negotiation to the identifying interests phase, the mediator thereby makes available some new capabilities, most principally the ability for users to contribute interest items. To contribute interest items, users click on the Interest Items link. After composing an interest item in the text box that then appears, and clicking on Add Interest Item (Figure 7), the user's interest will be made visible to all users and the mediator, along with the name and party affiliation



of the contributing user. During this phase, the group issue statement is available to the users for reference on the top of the screen, along with the mediator's summary of how the negotiation seems to be going or what problems the phase is currently addressing. As will be seen shortly, it is expected that different negotiations might be best facilitated if some degree of anonymity is guaranteed to the users who contribute interest items. STORM2 can be configured to support different degrees of anonymity, as shall be described shortly.

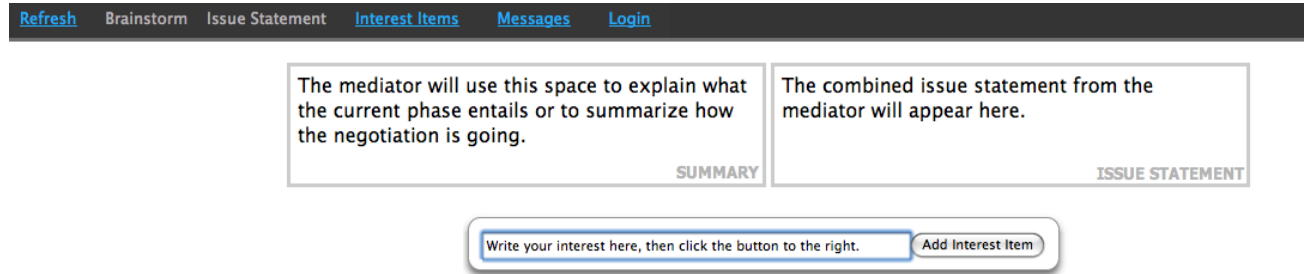


Figure 7: Identifying interests.

### 3.4 Options

After both parties have identified their interests, the mediator is then expected to move the negotiation to the phase where options are solicited. This is typically the longest phase of the negotiation. The phase is centered on the use of a brainstorming component that solicits initial options, then clarifying questions, and then further options. During this phase, users are asked to contribute options, or different ideas that may resolve some facet of the issue or satisfy some of the interests. To access this feature, users click on the Brainstorm link. To contribute an option, a user writes an idea in the text box that appears, and then clicks the Add Option button (refer to Figure 8). Typically, the option will be made visible to all users and the mediator, along with the contributing user's name and party affiliation, although here too different process details, supporting different levels of anonymity, may be desirable and can be defined and enforced using STORM2. A question mark button is displayed next to each option, and is used to enable a user to ask questions about an option that may seem to be unclear. Typically this button will be deactivated (users will not be able to click on it) until the mediator chooses to allow this type of interaction. But here again, different conventions for the activation of this button may be desirable, and can be supported by STORM2. During this phase, the group issue statement is available on the top of the screen for users to reference, along with the list of interests identified in the previous phase, and a summary from the mediator of how the brainstorm is going or what problems the phase is currently addressing.

### 3.5 Clarifying Questions

When participants are done contributing options (there are a number of ways in which the criteria for determining this might be defined used STORM2), the mediator is then expected to proceed to the next phase, allowing participants to ask clarifying questions. Users are typically still able to add new options during this phase, in case a question gives a user a new idea. To ask a clarifying question about an option, a user can click on the question mark button next to that option, write a question in the text box in the dialog that pops up, and then click the Ask Question button (see Figure 9). The question will appear underneath the appropriate option, possibly (depending upon how the specific STORM2 process and user interface details are defined) along with the user's name and party affiliation. During this phase, the group issue statement, the interest items, and the mediator summary are still available for viewing.

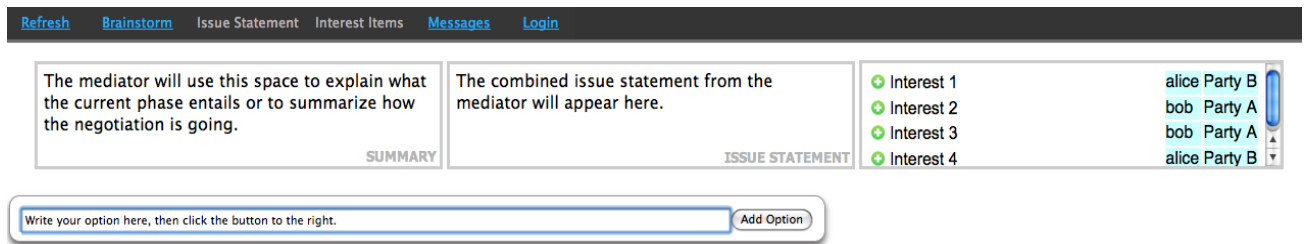


Figure 8: Contributing options.

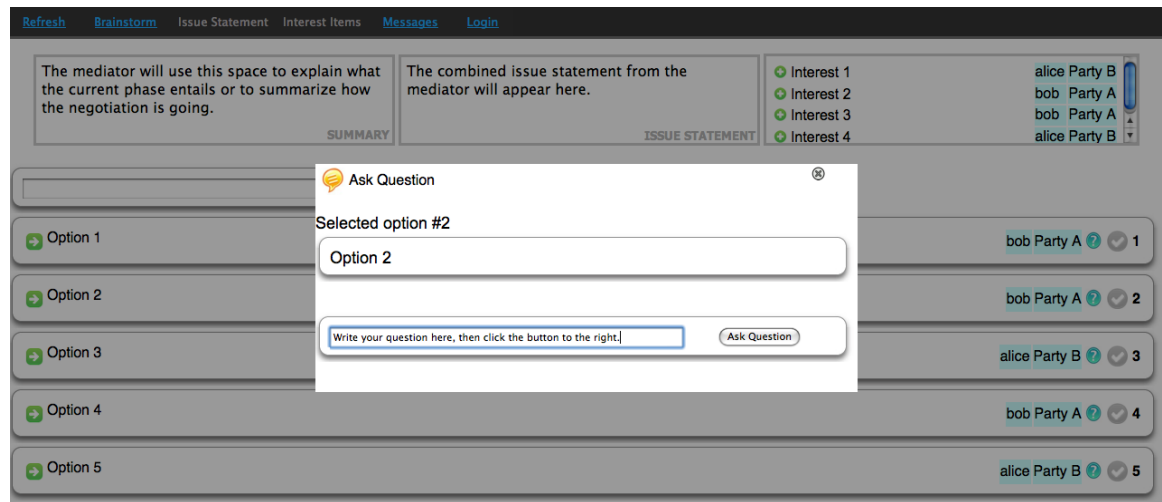


Figure 9: Asking clarifying questions.

The mediator may also answer questions. An answer will appear on users' screens as an indented line after a question, preceded by a yellow text bubble icon (Figure 10).



Figure 10: Answers.

### 3.6 Mutually acceptable options

The last phase of the structured negotiation that STORM2 supports is identifying options that are acceptable to both parties. When the defined criteria for the specific STORM2 process determines that the participants are done brainstorming, the mediator will then change the phase to caucus. During caucus, the participants typically will not have access to any functionality except the message center in case they need to contact the mediator, and thus the caucus is effectively a pause in the process after the brainstorming phase of the negotiation. During this phase, party representatives are expected to discuss the contributions with their respective teams and work on identifying acceptable options that the party representative should vote for in the final phase of the negotiation process. These mutually acceptable options can then be used to formulate a resolution. This functionality is typically only available to the party representatives; however, if a user is not a party representative, he or she typically will still be able to view what options his or her party representative has voted for, but will not be able to make any changes in these votes. This functionality is available under the Brainstorm link. There is a gray check mark next to each option, which party representatives can click to identify the option for which the representative is voting. Upon voting, the option's corresponding checkmark will turn green (Figure 11). To uncheck an option checked by mistake, the party representative can click on its checkmark again and it will turn gray. Different choices of permissions that might be given to various participants are possible and can be defined using STORM2 process definition facilities.

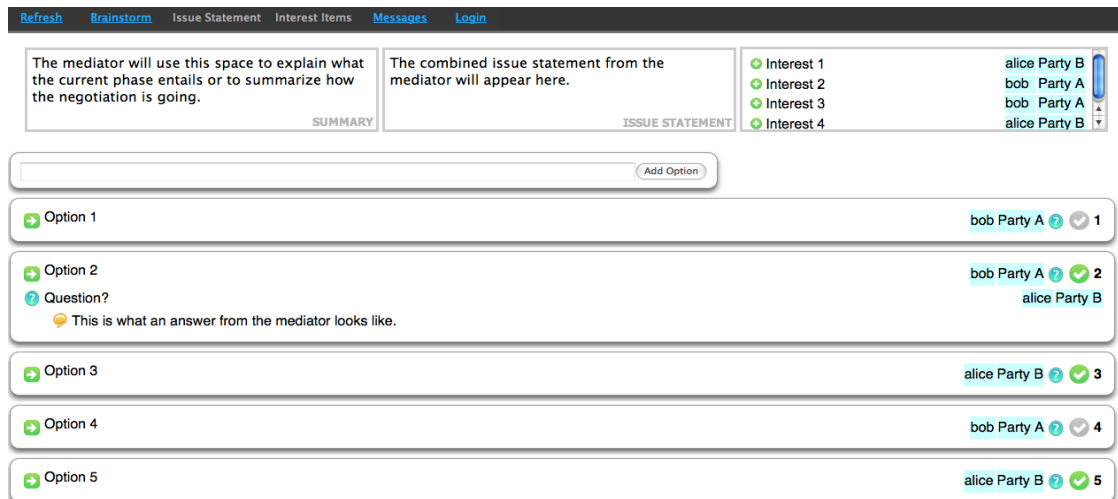


Figure 11: Identify acceptable options.

The preceding discussion has suggested ways in which STORM2's process can be tailored so that the identity of the different participants in the negotiation can be concealed. For example, high animosity between the negotiating parties may suggest the desirability of anonymous brainstorming where the users are not aware of who made what contributions. STORM2 supports such variation in degree of visibility by providing different anonymity settings. At present the anonymity setting choices are full attribution (author name and party affiliation are displayed), party attribution (the author name is anonymized, but the party affiliation is displayed), and full anonymity (all contributions' author fields appear as anonymous, with no party affiliation). These anonymity settings are controlled by the mediator and are fully dynamic, meaning that the mediator can switch the level of anonymity at any point during a negotiation. This flexibility is important as mediators may wish to start with a fully anonymous negotiation and later move to an attributed negotiation, or conversely.

The details of how the different levels of anonymity are effected are described in the following section. Anonymity is only one example of different variations that arise in the dispute resolution process and that ODR systems should be able to manage. As noted above other forms of tailoring might be used, for example, to adjust the criteria used to determine when the different phases are to be ended. STORM2 can easily be modified to accommodate such variations in process detail as the need for them may arise.

## 4 Implementation and Experience

STORM2's implementation architecture can be conceived as being comprised of three main components, namely the process that guides the negotiation, a web-enabled graphical user interface for the different participants in the negotiation, and the Dispatcher system that, through its binding script component, passes information and commands between the process and the user interface.

The process for guiding the negotiation is, as described above, implemented as a process definition in the Little-JIL language. Details of this language are beyond the scope of this paper, but it should be noted that the language, much like a programming language, has rigorously defined, unambiguous semantics and is executable. Thus, once a process has been defined in Little-JIL it can be executed directly. Further it is important to note that the language is visual, using icons to diagrammatically represent the coordination model of defined processes. This seems to make Little-JIL process definitions relatively more accessible to domain experts, thereby inviting their greater participation in the definition and possible modifications to negotiation processes. In addition to the coordination diagram, the process has an artifact specification as outlined in Section 2. For the process that guides STORM2, this artifact specification includes a repository that maintains and manages the current state of the negotiation (e.g. the contributed interest items, options, and questions). At present, this repository is implemented by a simple in-memory data structure with tables for the different contributions. As the state of even a complex negotiation is not expected to ever become very large, it is expected that this implementation choice will remain sufficient for the foreseeable future.

The user interface component is actually a composite of interfaces for each of the different participants and the mediator. These interfaces are responsible for presenting requests for user participation when the process definition indicates that this is required, and for capturing the inputs provided by users. Communications with various users (the disputants and the mediator) are effected with the help of a software component called the Dispatcher, which intermediates the exchange of information between the process and its participants' user interfaces in a manner that is defined precisely by a specification component called a binding script, as indicated in Figure 1. The Dispatcher middleware component seamlessly controls all user interfaces based on the current state of the process execution. For example, it is responsible for passing inputs acquired from a participant through the participant's interface to the process so that these inputs can be stored in the process artifact repository and can then subsequently be distributed to other users as specified by the process definition. The Dispatcher thus supports all communication of information and passing of commands between the process and the users of the STORM2 system. In ODR processes this information typically consists of options, questions, etc., as well as information that identifies the source of this information.

The choices of architecting these three components have been carefully considered and implemented with the goal of providing as much flexibility as possible. In particular, by establishing these three separate areas of responsibility it is expected that changes to each can be made with minimal requirement for making changes to any of the other components.

Additionally, the choice of the Little-JIL process definition language for the realization of the backbone of the STORM2 system, namely the negotiation process, provides the advantage of a high-level design language, so changes to the coordination of participants or the sequence of different activities are not only confined to the process component, but can often be carried out as simple manipulations of the diagrammatical representation of the process, without further programming being necessary. This reduces the possibility of introducing errors when making changes and helps to keep the domain experts more involved in the definition and refinement of the negotiation process.

Similarly, we have implemented such features as anonymity control fully within the process without affecting the Dispatcher or the user interface, by embedding an anonymity control module within the process definition, where it is used to decide which identification information is to accompany user contributions when displayed to users on their screens. We note that all information that is stored in the repository is always fully attributed, but that the attribution that is actually seen is determined solely by the process’s anonymity control module. This feature has proven to be quite effective in supporting a wide range of desired anonymity specifications, and in particular for supporting dynamic changes in anonymity settings as the process is executing. Anonymity control within STORM2 is explained in more detail below.

## 4.1 User Interfaces and Anonymity

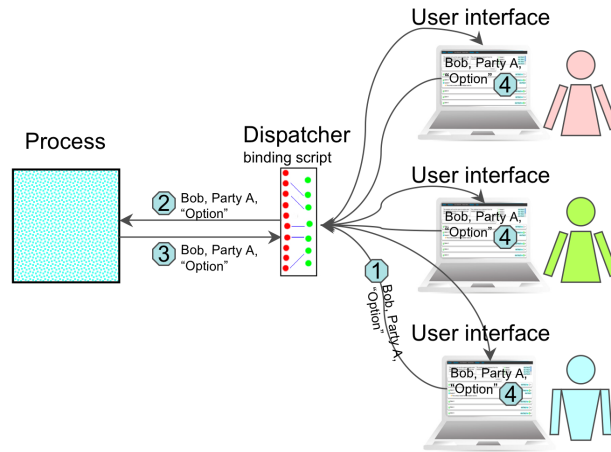


Figure 12: An example of a fully-attributed variant of the process.

Our belief is that in many cases it is important for participants in the process to know the sources of various items of information, but that, as noted above, there are some cases in which it might be better to conceal the identity of the originator of certain items of information.

Consider, for example, that a mediator decides that the authorship information should be visible to all participants. If participant Bob from Party A adds an option, it will go through several steps before it is displayed to the other participants, indicated by the numbered hexagons in Figure 12. First, Bob will compose his option and submit it through the user interface he is interacting through, effectively sending it to the Dispatcher. The Dispatcher will then pass on all this information, namely the author, party affiliation, and option text, to the executing STORM2 process. The process will then handle this input appropriately, and, since the mediator has indicated that users should be able to see all the authorship information, no information will be redacted out before the process hands the option back to the Dispatcher to be displayed to other users through their user interfaces. Thus step 4 shows the Dispatcher passing the option to the users’ interfaces, which then display “Option” from participant Bob from Party A on the users’ screens.

On the other hand, consider a process variation in which the mediator has requested that all information be anonymized. In this case a contribution will go through the same steps, but with very different results. This new variation is illustrated in Figure 13. The option from Bob of Party A will follow the same path until step 2 where the process deals with the newly submitted option. The process will now have been instructed by the mediator to remove the author and party information

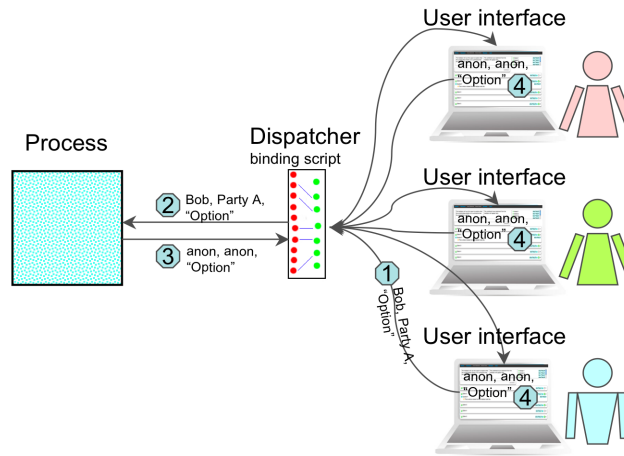


Figure 13: An example of a fully anonymous variant of the process.

of any contribution that passes through. Therefore, in step 3, the process will hand over to the Dispatcher a redacted option, consisting of the text “Option,” but without any specification of the name or party affiliation of the author. Consequently, in step 4, the users will see a new option, “Option,” on their screens but the identity of the contributor of this option will not be shown.

In Figures 12 and 13, it is obvious that the process is responsible for the anonymization of data, but it is not clear how that anonymization is carried out. In order to support the ability to either conceal or reveal the source of items of information in a STORM2 process, an anonymization control module called the Redactor agent is used. The Redactor is an automated support tool, or a software agent, that is defined within the process’s agent specification, as outlined in Section 2. Its function is to modify the content of messages passed from the process to the Dispatcher in accordance with explicit specifications. Usually these modifications consist of concealing the identity of the source of information. Commands that configure the Redactor’s anonymization activities are generated during the execution of STORM2 processes, thereby enabling the fine-grained control of just what kinds of anonymization should take place during what parts of the process. While Figures 12 and 13 illustrate how different levels of anonymity are achieved in the STORM2 system at a very high level, Figure 14 explains the anonymity control mechanism and details where the Redactor fits within the process and the overall STORM2 architecture.

Figure 14 is an elaboration of the Dispatcher’s interaction with the process from Figures 12 and 13 and details the case when anonymization is required. After the Dispatcher has handed an artifact from the user’s interface to the process in step 2, namely an option from Bob from Party A, the artifact then goes through several steps within the process before it is passed on to other users. The process steps specify that the option that it has received from the Dispatcher be stored in the process’s options repository as shown in step 2.1. The options repository resides within the artifact specification module of the process definition, along with other repositories for interests, questions, and so on (not shown). In order for participants to see artifacts such as the list of options, the process must give these artifacts to the Dispatcher to be passed on to the interfaces to the relevant users. In order for this to happen, however, first (as shown in step 2.2) a copy of the artifact (denoted by a red C) must be taken from the artifact repository, and passed to the Redactor agent module. Note that since the options repository is handing over a copy of the option and not the actual artifact, full information about the option is retained should it be needed in the future. The Redactor then redacts the copy of the option as authorized, in this

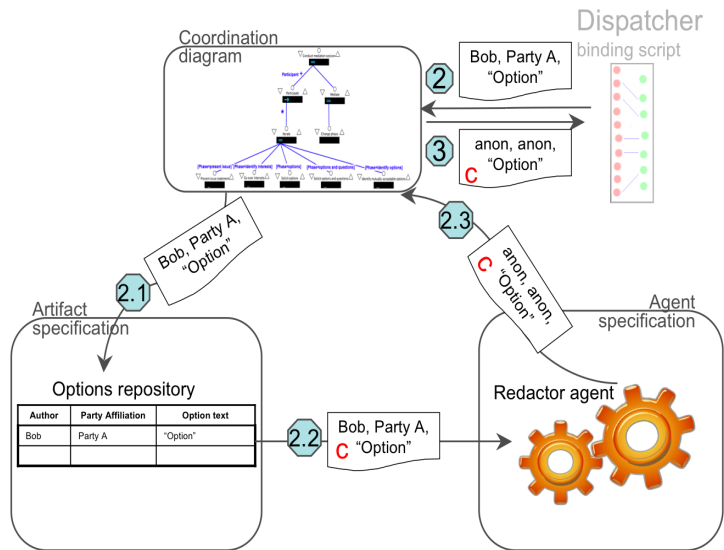


Figure 14: A Little-JIL process comprises three elements, which work together to effect information redaction.

case removing the author and party information, and thus makes the redacted artifact available (step 2.3 in the diagram). Finally, the copy of the redacted option (indicated by the red C) is handed back to the Dispatcher in step 3.

It is important to note that the mediator may decide that anonymity is valuable, for example, during some intervals of the brainstorming subprocess, but that anonymity may not be desirable at a later period. STORM2 allows for dynamically switching the behavior of the Redactor software agent to support such situations. In addition STORM2 also supports different levels of anonymity and attribution. Thus STORM2 allows editing out all information about the author, but also allows editing out only the name of an author, while retaining the author's party affiliation. In some preliminary evaluations the STORM2 experimental platform has been used to explore the value of different levels of anonymity at different times during different interest-based negotiations. Some preliminary results of this experimentation are summarized below.

## 4.2 Experience and Early Evaluation

Exploration of the value of different anonymity settings has been an early focus of our evaluation of STORM2. This seems particularly important since historically, anonymity has not been possible in face-to-face negotiations. STORM2 has been shown to be appropriate for carrying out social science research because the system's built-in process facilitates replicability among different studies and the strict process control ensures that human subjects do not stray from the current task at hand, reducing variability among participants. In these preliminary studies, eight participants used the system to negotiate two different case studies, one fully attributed and one fully anonymous. A neutral third party served as the mediator in both case studies. These case studies aimed to explore the effect of anonymity on several different facets of user behavior, and to shed light on the following research questions: does increased anonymity result in: increased participation (more submissions per user), increased inclusion (submissions from more users), increased empowerment (more users feel heard), increased equality in participation (resulting in comparable number of submissions from users not in a position of power as ones in a perceived position of power), and increased satisfaction (more users report positive

experience)?

In these preliminary studies, anonymity had a clear impact on these different aspects of user participation and satisfaction. Most notably, the majority of participants reported feeling a greater freedom of expression when the negotiations were anonymous, and most participants also reported less self-censoring and fear of retaliation in the anonymous case study compared to the attributed case study. Interestingly, most participants raised concerns about how knowledge about authorship may affect their judgement, indicating that they were more likely to judge an idea simply on its merit when they were unaware of who the author was. Additionally, the anonymous case study resulted in more contributions deemed acceptable by both parties, both in terms of their number, as well as their percentage of the total pool of options. These exploratory studies and their findings suggest that different levels of anonymity in ODR may be valuable for certain situations and encourage further investigation. More case studies are currently under development to further explore these preliminary findings.

The STORM2 system seemed to be very effective in supporting experiments. Because of its explicit process support, it was able to provide a level of rigidity and structure that is not usually available in off-the-shelf ODR software, enabling the conduct of multiple case studies in a very similar, structured fashion. This is encouraging because it indicates that STORM2 may be able to mitigate one of the challenges of conducting multiple sessions with human participants—namely, reproducibility. By providing explicit process support and control, STORM2 can reduce the variability that can be introduced by a software system that does not limit the participation of subjects based on the current phase of the negotiation, for example. STORM2 has been able to support one aspect of variation, namely different anonymity settings, very well. Additional work is necessary to study the effectiveness of its architecture in supporting different modalities of negotiation and further dimensions of variability.

Finally, we note that the Little-JIL process definition language has been specifically devised to facilitate making changes to process definitions. As noted above the fact that Little-JIL is a visual language seems to make process definitions relatively more accessible to negotiation domain experts. The Visual-JIL process definition editor has been used to support a variety of changes to process definitions. Little-JIL is a hierarchical decomposition language, and so changes to lower level details seem to be particularly easy to make, often entailing the addition, deletion, or modification of process features at the lower levels of the process hierarchy. There have, however, been cases in which process changes have required substantial modification to the process definition. In some cases, the magnitude of required changes was surprising, and seemed inconsistent with the perceived difficulty of the changes.

## 5 Future Work

Our experience with STORM2 to date is still quite preliminary. As noted in the previous section, it seems that the architecture of STORM2 and the implementation of the three principal modules is facilitating the flexibility that we believe is quite important. But considerable additional work is needed to determine whether the flexibility is adequate, and determine whether changes, either to the architecture or to any of the principal components, must be made to effect desired improvements.

Thus, for example, one ongoing research direction is the exploration of the anonymity control features of STORM2 and their effect on negotiations. Preliminary experience is suggesting that the flexibility that has been incorporated is welcome and useful. On the other hand, the availability of this flexibility in anonymity control is suggesting the possibility of approaches to negotiation that had been difficult or impossible previously. Thus, STORM2 is suggesting the possibility of negotiation approaches that may yet highlight the need for system capabilities that are not yet incorporated. Ongoing research will clarify this.

In addition, it seems important to explore the extent to which the process definition language can support still broader types of negotiation processes. Note that this report focused on a single paradigm for negotiating, namely the interest-based mediation model. But there are other modalities for conflict resolution that are also effective and widely used, and



the choice of one modality over another is often the result of cultural preferences. Although the interest-based mediation model may be the most widely used in North America, some cultures may find models that are based on relationship building to be more effective. We would like to explore and define different negotiation process variants that model such different paradigms, by focusing on building relationships instead of solving problems, for example. Such process variants would likely be of value to the conflict resolution community, and they would also help us to evaluate the current techniques we have identified for accommodating variation in processes, and may elucidate new techniques. As a result, we envision that STORM2 will continue to evolve and provide support for multiple process variants, so that eventually it can support a whole family of related processes for different mediation situations. We could even hypothesize dynamically switching between such related processes in the future, if a negotiation situation necessitates it.

This experimentation with different negotiation processes will also help us to evaluate the expressive power of Little-JIL. Early experimentation suggested that Little-JIL was able to support specification of a gratifying range of different negotiation processes. On the other hand, as noted above, some process changes turned out to be harder to make than was expected. Indeed, the need to make these changes led to deeper understandings of the process architectures that seem most useful in supporting changes of different types. Changes to actual language features of the Little-JIL process definition language may also be indicated as our experimentation continues.

Finally, all of this experimentation will also shed light upon the efficacy of the STORM2 architecture. We expected that the division of the system into the three principal components would help to orthogonalize the key system responsibilities in a way that would facilitate very broad types of modifications. Continued experimentation will provide the basis for supporting evaluation of these architectural decisions.

## 6 Related Work

There are many software systems for collaboration and online brainstorming and dispute resolution that appear similar to STORM2. In general, such systems can be classified as either content-collaboration systems, domain-specific technologies, or toolkits for groupware development. For example, LotusNotes [8] is a member of the third category and can support some process-guidance with an explicit embedded process programmed in advance. LotusNotes is strictly a groupware system, however, with no built-in features for guiding dispute resolution. Furthermore, such predefined processes are not as flexible and evolvable as the process definition in STORM2. Collaborations systems usually provide project collaboration (e.g. document sharing and editing) and conferencing capabilities, such as WebEx [4], Central Desktop [3], Groove [9], and SameTime [7], or explicit brainstorming capabilities, such as MindMeister [10] and Caucus [2]. These systems do not provide any process guidance. Although such systems can be adapted for ODR by experienced mediators, they are often cumbersome to use as this was not their intended purpose, and moreover they do not provide any explicit support for the interest-based mediation process. Thus, they are usually used for only part of a negotiation session, or in combination with other systems, introducing additional complexity, and a steep learning curve for new mediators.

There are also domain specific technologies built for ODR, such as the ThinkTank collaboration technology [13], Facilitate.com [6], and The Mediation Room [12], that support group decision making. Such systems can implicitly accommodate some variation by explicit parameterization during initial setup, such as choosing an anonymity setting. The STORM2 system, with its explicit process guidance and built-in variation support for dynamic changes to anonymity offers clear advantages to such approaches.

Neither the “process-agnostic” collaboration systems nor the domain-specific tools support explicit process representation, or a specification of a process to be executed. Although some support exists for parameterization and thus supporting a number of related processes, or variants, these systems do not foster understanding of process variation or the inherent relationships among process variants. Moreover, since there exists no explicit process, the disputants and participants cannot benefit from the sort of guidance that STORM2 can provide based on its underlying process definition, or from the ease of use encouraged by STORM2’s support of only the features necessary to support interest-based mediation, without any

extra or unnecessary features. On the other hand, groupware construction tools can support the explicit specification of one rigid process but cannot provide support for representing variations of multiple variants, and cannot therefore encourage reasoning about variation.

We share Briggs's view that research in collaboration should not focus on "collaboration technology," but should be centered around "technology supported collaboration processes" instead [19]. Thus we believe that it is important to focus on how best to support process definitions that are powerful yet flexible. There are several approaches to modeling and representing processes. Some use different mathematical formalisms, such as PetriNets [24, 28] or finite state machines [29, 32], while others are based on the extension of models of the process similar to a flowgraph [15, 36, 46]. Generally, these approaches concentrate on the workflow, or coordination of activities, while neglecting other aspects that are necessary for successfully modeling a process holistically [47], such as artifacts, agents, and resources. Agents and resources are often neglected, and when modeled [18, 23, 46] the models are usually simplistic. While some modeling techniques recognize the need for artifacts [43, 46], they are often modeled by means of simple type systems. These shortcomings seem to be addressed by approaches such as ours that treat process definitions as a form of software, thus treating data artifact specifications as being necessary complements to activity specifications, and then borrowing from the literature and technology of programming to support process definition.

Indeed it is our view that processes can be defined and represented in a way that is similar to general software system modeling approaches [39, 40]. Although initial efforts concentrated on modeling the software development process, much work has since been done on modeling general workflows [46, 35], as well as processes in domains as diverse as business [27, 44], service-oriented architectures [14, 26], science [15, 38], elections [42], and medicine [21, 30], as well as this work on dispute resolution and mediation. Among the advantages of this proposed approach is the opportunity that it seems to present for dealing with process variation [41].

The work we describe here represents an important continuation of ongoing efforts to explore vehicles for facilitating process variation. Parallels can be drawn between accommodating variation in the process domain and work on software product lines and software families [37]. Although little has been done to study variability and variation accommodation within processes, our work is similar to the notion of using feature models or initial configuration specifications and then applying generation techniques for parameterization within the software domain [17, 22, 33]. Domain-specific feature graphs for specifying variability within a model, such as FORM [34] are conceptually very similar to accommodating process variations, however, the explicit enumeration of all possible mandatory and optional features in such approaches can lead to very large feature graphs, as contrasted to our proposal of only including the desired features. Similarly, decision models (e.g. Kobra [16] and FAST [45]) can be used for instance generation and variant modeling, where variation points are indicated as decisions and, based on the selection, the model is extended with different sets of features.

We feel that STORM2 fills an important niche in the world of ODR technology by providing much needed explicit process definitions, opportunities for customization and evolution of these definitions, with greater facility than is available with existing systems, and current support for on-the-fly changes to anonymity settings that have, until now, been available only as statically-defined settings. Additionally, unlike other work that tries to provide an all-in-one solution, where too many features may actually impede the mediator, STORM2 provides only the features necessary for performing interest-based mediation effectively.

## 7 Conclusion

In conclusion, STORM2 is an online system for dispute resolution which provides important process guidance. The architecture of the system encourages and facilitates evolvability and variability, which makes STORM2 very flexible and well poised for addressing some of the challenges that the online dispute resolution domain presents. We believe that this flexibility makes STORM2 a suitable vehicle for exploring a multitude of software engineering as well as social science research avenues. STORM2 has been used as the experimental platform in a set of case studies exploring the social impact

of anonymity on the behavior and satisfaction of participants in negotiations, as well as the suitability of STORM2 in accommodating the variation necessary for supporting the different levels of anonymity with promising results.

In preliminary testing, STORM2 has been very well liked by negotiation professionals. Mediators seem to find the system easy and intuitive to use because it follows the interest-based negotiation process with which most North American mediators are already familiar, and because it includes few, if any, extraneous features. In addition, mediators appreciate the control they have over how quickly or slowly the process moves forward, as well as the flexibility to control anonymity dynamically depending on group dynamics and involvement. The STORM2 system has also provided the basis for a useful case study, exploring different software engineering issues arising from trying to apply technology to the field of dispute resolution and negotiation. In addition, it has also proven to be a suitable vehicle for trying to assess some of the other challenges that ODR faces, such as gauging the impact of anonymity.

## 8 Acknowledgements

The authors thank Matt Marzilli, Lori Clarke, Alexander Wise, Leah Wing, Alan Gaitenby, Ethan Katsh, and Norm Sondheimer of the University of Massachusetts, Amherst, as well as Daniel Rainey of the National Mediation Board for their insightful input. This research was supported in part by the U.S. National Mediation Board, and in part by the U.S. National Science Foundation under Award Nos. CCR-0427071, CCR-0204321, and CCR-0205575. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied of the U.S. National Mediation Board, or the U.S. National Science Foundation, or the U.S. Government.

## References

- [1] Apache Tapestry. <http://tapestry.apache.org/>.
- [2] Caucus. <http://www.caucuscare.com/>.
- [3] Central Desktop. <http://www.centraldesktop.com/>.
- [4] Cisco WebEx. <http://www.webex.com/>.
- [5] eBay Resolution Center. <http://resolutioncenter.ebay.com/>.
- [6] Facilitate.com. <http://facilitate.com/>.
- [7] IBM Lotus SameTime. [www.ibm.com/sametime/](http://www.ibm.com/sametime/).
- [8] Lotus Notes by IBM. [www.ibm.com/software/lotus/products/notes/](http://www.ibm.com/software/lotus/products/notes/).
- [9] Microsoft Groove. [www.groove.net](http://www.groove.net).
- [10] MindMeister. <http://www.mindmeister.com/>.
- [11] PayPal Dispute Resolution. <http://www.paypal.com/cgi-bin/webscr?cmd=xpt/cps/general/PPDisputeResolution-outside>.
- [12] The Mediation Room. <http://themediationroom.com/>.

- [13] ThinkTank by GroupSystems. <http://www.groupsystems.com/thinktank>.
- [14] ALONSO, G., AGRAWAL, D., ABBADI, A., KAMATH, M., GUNTHOR, R., AND MOHAN, C. Advanced transaction model in workflow context. In *Proceedings of the 12th IEEE International Conference on Data Engineering, Proc. 12th International Conference on Data Engineering, New Orleans, February 1996* (1996), pp. 574–581.
- [15] ALTINTAS, I., BERKELEY, C., JAEGER, E., JONES, M., LUDÄSCHER, B., AND MOCK, S. Kepler: An extensible system for design and execution of scientific workflows. In *Proceedings of the 16th International Conference on Scientific and Statistical Database Management, Santorini Island, Greece* (2004), pp. 423–424.
- [16] ATKINSON, C., BAYER, J., AND MUTHIG, D. Component-based product line development: The KobrA approach. In *Proceedings of the The First International Software Product Line Conference, Denver, CO* (2000), pp. 289–309.
- [17] BATORY, D., AND O’MALLEY, S. The design and implementation of hierarchical software systems with reusable components. *ACM Transactions on Software Engineering and Methodology* 1, 4 (1992), 355–398.
- [18] BELKHATIR, N., ESTUBLIER, J., AND WALCELIO, M. ADELE-TEMPO: an environment to support process modelling and enactment. In *Software Process Modeling and Technology* (1994), pp. 187–222.
- [19] BRIGGS, R. O. On theory-driven design and deployment of collaboration systems. *Int. J. Hum.-Comput. Stud.* 64, 7 (2006), 573–582.
- [20] CASS, A., LERNER, B., SUTTON JR, S., MCCALL, E., WISE, A., AND OSTERWEIL, L. Little-JIL/Juliette: A process definition language and interpreter. In *Proceedings of the 22nd International Conference on Software Engineering, Limerick, Ireland* (2000), pp. 754–757.
- [21] CLARKE, L., CHEN, Y., AVRUNIN, G., CHEN, B., COBLEIGH, R., FREDERICK, K., HENNEMAN, E., AND OSTERWEIL, L. Process programming to support medical safety: A case study on blood transfusion. In *Proceedings of the Software Process Workshop 2005, Beijing, China* (2005), Springer-Verlag, pp. 347–359.
- [22] CZARNECKI, K., AND EISENECKER, U. Components and generative programming. In *ESEC/FSE-7: Proceedings of the 7th European software engineering conference held jointly with the 7th ACM SIGSOFT international symposium on Foundations of software engineering, Toulouse, France* (1999), pp. 2–19.
- [23] DAMI, S., ESTUBLIER, J., AND AMIOUR, M. APEL: A graphical yet executable formalism for process modeling. *Automated Software Engineering: An International Journal* 5, 1 (January 1998), 61–69.
- [24] EMMERICH, W., AND GRUHN, V. FUNSOFT nets: a petri-net based software process modeling language. In *IWSSD '91: Proceedings of the 6th International Workshop on Software Specification and Design, Como, Italy* (1991), pp. 175–184.
- [25] FISHER, R., URY, W. L., AND PATTON, B. *Getting to Yes: Negotiating Agreement Without Giving In*, 2 ed. Penguin, 1992.
- [26] FOSTER, H., UCHITEL, S., MAGEE, J., KRAMER, J., AND HU, M. Using a rigorous approach for engineering web service compositions: A case study. In *SCC '05: Proceedings of the 2005 IEEE International Conference on Services Computing* (2005), pp. 217–224.
- [27] GEORGAKOPOULOS, D., HORNICK, M. F., AND SHETH, A. P. An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and Parallel Databases* 3, 2 (1995), 119–153.

- [28] GHEZZI, C., MANDRIOLI, D., MORASCA, S., AND PEZZE, M. A unified high-level petri net formalism for time-critical systems. *IEEE Transactions of Software Engineering* 17, 2 (1991), 160–172.
- [29] HAREL, D., AND NAAMAD, A. The STATEMATE semantics of statecharts. *ACM Transactions on Software Engineering and Methodology* 5, 4 (1996), 293–333.
- [30] HENNEMAN, E., COBLEIGH, R., FREDERICK, K., KATZ-BASSET, E., AVRUNIN, G., CLARKE, L., OSTERWEIL, J., ANDRZEJEWSKI, C., MERRIGAN, K., AND HENNEMAN, P. Increasing patient safety and efficiency in transfusion therapy using formal process definitions. Tech. rep., University of Massachusetts, Amherst, 2006.
- [31] KATSH, E., AND RIFKIN, J. *Online Dispute Resolution: Resolving Disputes in Cyberspace*. John Wiley & Sons, 2001.
- [32] KELLNER, M. Software process modeling support for management planning and control. In *Proceedings of the First International Conference on the Software Process, Redondo Beach, CA* (1991), pp. 8–28.
- [33] KNAUBER, S. Synergy between component-based and generative approaches. In *ESEC/FSE-7: Proceedings of the 7th European Software Engineering Conference Held Jointly with the 7th ACM SIGSOFT International Symposium on Foundations of Software Engineering, Toulouse, France* (1999), pp. 2–19.
- [34] KYO, C., KANG, S., LEE, J., KIM, K., SHIN, E., AND HUH, M. *FORM: A Feature-Oriented Reuse Method with Domain Specific Reference Architectures*. No. 5 in *Annals of Software Engineering*. pp. 143–168.
- [35] LEYMANN, F., AND ROLLER, D. Workflow-based applications. *IBM Systems Journal* 36, 1, 102–123.
- [36] MAYER, R., AND ET AL. IDEF family of methods for concurrent engineering and business re-engineering applications. Tech. rep., Knowledge Based Systems, Inc., 1992.
- [37] NORTHROP, L. *Software Product Lines—Practices and Patterns*. Addison-Wesley, 2002.
- [38] OSTERWEIL, L. Software processes are software, too, revisited. In *Proceedings of the 19th International Conference on Software Engineering, Boston, MA* (1997), pp. 540–558.
- [39] OSTERWEIL, L. J. Software processes are software too. In *9th International Conference on Software Engineering (ICSE 1987)* (Mar. 1987), pp. 2–13.
- [40] OSTERWEIL, L. J. Software processes are software too, revisited. In *19th International Conference on Software Engineering (ICSE 1997)* (Sept. 1997), pp. 540–548.
- [41] SIMIDCHIEVA, B. I., CLARKE, L. A., AND OSTERWEIL, L. J. Representing process variation with a process family. In *Software Process Dynamics and Agility: Proceedings of the International Conference on Software Process* (2007), Q. Wang, D. Pfahl, and D. M. Raffo, Eds., vol. 4470 of *LNCS*, Springer, pp. 109–120.
- [42] SIMIDCHIEVA, B. I., MARZILLI, M. S., CLARKE, L. A., AND OSTERWEIL, L. J. Specifying and verifying requirements for election processes. In *dg.o '08: Proceedings of the 9th Annual International Conference on Digital Government Research* (2008), S. A. Chun, M. Janssen, and J. R. Gil-Garcia, Eds., Digital Government Society of North America, pp. 63–72.
- [43] SUZUKI, M., AND KATAYAMA, T. Meta-operations in the process model HFSP for the dynamics and flexibility of software processes. In *Proceedings of the First International Conference on the Software Process, Redondo Beach, CA* (1991), IEEE Computer Society Press, pp. 202–217.

- [44] WEIGERT, O. *Business Process Modeling and Workflow Definition with UML*. 1998.
- [45] WEISS, D. M., AND LAI, C. T. R. *Software product-line engineering: a family-based software development process*. Addison-Wesley, 1999.
- [46] WHITE, S. A. Business process modeling notation (bpmn). Tech. Rep. formal/2009-01-03, Business Process Management Initiative (BPMI), January 2009. Version 1.2.
- [47] ZHU, L., OSTERWEIL, L. J., STAPLES, M., KANNENGIESSER, U., AND SIMIDCHIEVA, B. I. Desiderata for languages to be used in the definition of reference business processes. *International Journal of Software and Informatics* 1, 1 (December 2007), 37–65.